BBS Express! Professional

ExpressNET! Networking

By Keith Ledbetter and Chris King

Orion Micro Systems
2211 Planters Row Drive
Midlothian, Virginia 23113

For Technical Assistance, call one of our support boards.

BBS PRO Support Board (804) 744-9987
Midnight Express BBS (804) 379-4156
M.O.U.S.E. BBS (219) 674-9288
Network Atari BBS (512) 662-9765
Ol' Hackers BBS (515) 884-4140

Printing History:
1st Printing Edition - *November 1988*

0104

BBS Express! Professional

ExpressNET! Networking

User Manual

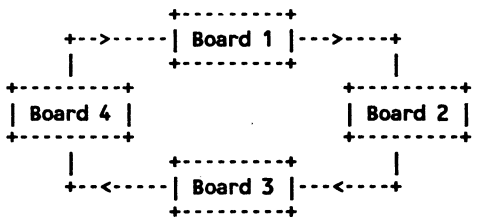
ExpressNet! Networking

Disclaimer: Orion Micro Systems is in no way liable for any phone bills incurred while using the ***ExpressNET!*** system, due either to owner mis-use or program errors.

The ***ExpressNET!*** networking feature of PRO is an automatic message networking facility for exchanging messages with another BBS Express! PRO system. This can be a second board you run, or a PRO board across town or across the country.

Operationally (from the sysop's point of view) it is a very simple process.

Here is a pictorial of a sample network setup.



As you can see, this sample network contains 4 boards. The arrows show the direction the messages would flow. The key thing to remember when networking is 'the board initiating the call receives the messages'. In order to achieve this flow, board 2 would call board 1 to receive new messages. Board 3 calls board 2. Board 4 calls board 3. And to complete the loop, board 1 calls board 4.

In a 2 board network, board 1 would call board 2 to receive messages and then board 2 would call board 1 to receive messages. This was done this way, so that 1 board would not have to bear all the cost of long distance charges. After all, why should you pay the LD costs for someone else to get messages to post on their board. You make the call, you get the messages, simple as that!

Networking with another PRO board is not something that *'just happens'*. You and the sysop you will be networking with must both agree on the bases you want transferred and set up each side accordingly. You must set up a user id for the other sysop's use when calling your board as a network caller, and he must set one up for you to use when you call his board thru ***ExpressNET!.*** Once you have this information, you are ready to set up the parameters necessary to network with the other board.

Network Module Overview

ECHONET.DAT contains the node list of boards to build outgoing message packets. This file resides in the network subdirectory and can be edited with any word processor. It contains one line for each board entry and looks like this:

```
1  25  PRO Support Board
|   |   |
|   |   Comments (Optional)
|   |
|   The nodes user number
|
The node number of board for which
you are building the message packet
```

NOTE: Each of these fields are separated by 1 space.

CALLOUT.DAT - contains the network node list of boards that you are calling to receive message packets. This file is created and maintained using the **NETEDIT.CMD** module.

INCOMING.DAT - this is the data file which contains the incoming message packet containing messages that will be posted on your board by NETUPDT. When NETUPDT posts the messages to your bases, it automatically deletes this dataset. This file is automatically created by NETMODEM.

NETEDIT.CMD - This module is used to edit your CALLOUT dataset. It is this dataset that tells NETCALL information about each board that you want to call.

NETPREP.CMD - This module is ran once daily (recommended) from your event scheduler. It builds the message packets for each node you specify in the ECHONET.DAT file.

NETCALL.CMD - This module makes any necessary calls to other ExpressNET! BBS's to retrieve messages to be placed on your board.

NETUPDT.CMD - This module is invoked automatically by NETCALL if a file is received. It updates your message bases with the new network messages received in the INCOMING.DAT file.

NETHOST.CMD - This module is spawned off by the WAITCALL processor to handle network calls. Its purpose in life is to handshake with the network caller, validate the node number that is calling against your userlog file, and then initiate the file transfer if a network packet is waiting for the calling node.

NETMODEM.CMD - This module is the xmodem transfer module used by all ExpressNET! programs.

Note: All **.Dat** files reside in the **PRO>NETWORK>** subdirectory. All **.CMD** files should be copied to your **PRO>COMMANDS>(A-Z)** subdirectory.

Let's discuss each .CMD command in detail.

NETEDIT.CMD - this module runs from the Dos Shell. It allows you to enter up to 50 different nodes you wish to network with (ie receive incoming messages). From the Dos Shell,

run the program by typing **NETEDIT**. A menu will appear for you to select the entry number that you wish to edit. Select an entry number and press return and the following screen will apear:

```
  [A]  BBS Name :
  [B]  Number   :
  [C]  Baud:                [D] Seconds to wait:
  [E]  Usernum/PW:  0/
  [F]  Days to call: S M T W T F S
                     n n n n n n n
  [G]  Call between hours  : 0 - 0
  [H]  Maximum # of retries: 0
  [I]      Last tried                      :  0/00/00
           Last Connected                  :  0/00/00
           Last packet recv'd              :  0/00/00
  [L]  Incoming Message Redirection
----+----1----+----2----+----3-  --> (incoming bases)
          11111111122222222233
12245678931234567890123456789 01  --> (your bases)
| |      |
| |      |--> incoming base 10 to base 3
| |--------> incoming base  3 to base 2
|----------> incoming base  1 to base 1

Q)uit, T)rash,  A-L):
```

[A] **BBS Name** -The name of the BBS you are calling. This is the name that will appear in the selection menu once this entry is saved.

[B] **Number** - The phone number of the BBS for this entry.

[C] **Baud** - The baud rate to use when calling this BBS.

[D] **Seconds to wait:** - number of seconds to wait for a connection before hanging up.

[E] **Usernum/PW** - The user number and password to use when logging onto this BBS.

[F] **Days to call** - The days of the week to call this BBS.

[G] **Call between hours** - The hours of the day to call this BBS.

[H] **Maximum # of retries** - The number of times to try dialing this BBS.

[I] **Last tried/Last Connected/Last packet recv'd** - These fields hold the last dates data for the BBS. If you select this option, you will be prompted if you really want to clear these dates. Normally, you will never change these fields. But let's say you have tried to call the BBS the maximum retries, but you want to retry again. You can use this option to reset the dates back to 0, and the next time the NETCALL module runs, it will try to dial this entry again.

[L] **Incoming Message Redirection** - This option is used to set the "redirection" of incoming messages. The bar (1-32) represents the message base the incoming message was in on the board your networking with. The numbers below the bar represent the "redirection" or "in which base you want this message posted" on your board.

[T]rash - will delete and reset the current displayed node entry to empty.

[**Q**]uit - returns you to the Dos Shell. If any editing was performed while NETEDIT ran, the changes are saved back to **NETEDIT.DAT** before returning control to Dos Shell.

NETPREP.CMD - this module runs from the event scheduler. **NETPREP** reads the **ECHONET.DAT** file and builds the "message packets" for each of the node numbers that you have specified as being able to receive messages from your board. The **NETPREP** command should be placed in the event scheduler to be run once a day. **NETPREP** simply reads the **ECHONET.DAT** file and creates a packet (or file) for each node in the **ECHONET** file. The "user number" specified in the **ECHONET.DAT** file is actually used to read the userlog record so that **NETPREP** can determine which messages are "new" for each node. The user record will be updated and rewritten after **NETPREP** runs. The message packet for each node is placed in the **PRO>NETWORK>** subdirectory using the following data set naming convention:

X_nn.DAT

"nn" will be the NODE number of the packet receiver. So, if this packet was built for node number 1, the dataset name would be X_1.Dat. The node number is derived from the first parameter on each line of the **ECHONET.DAT** file.

It is perfectly acceptable to run the **NETPREP** every day to build message packets for your networking node's, but only have the nodes call in for messages every other day or even once a week. If **NETPREP** runs and determines that a dataset already exists for a node, it simply appends the additional messages to the packet. Once the node calls in to receive a waiting message packet, the packet will be deleted automatically upon a successful transfer with the receiver.

Should you desire, you may also run the **NETPREP** module manually from the Dos Shell. Why? Well, suppose you just started networking with a new node and wanted to get the initial transfer done to make sure everything was working correctly before setting it up in AUTO mode. You could manually run **NETPREP**, and have the receiver call in to receive (either automatically or using the **XMODEM** command from the Dos Shell) their message packet.

NETCALL.CMD - this module runs after the event scheduler. When this module runs, it reads the **NETEDIT.DAT** file and starts calling each board in the list until it connects with the board or has tried to call the maximum retries specified. Once connected, the boards handshake and start the message packet transfer if a packet exists. Once the packet is received, the connection is terminated and the receiving board passes control to the **NETUPDT** module to process the message packet which has been saved as **INCOMING.DAT** in the network subdirectory.

If for some reason, you do not want the **NETCALL** to run after you or a user logs off the BBS, you may hold the **START** key down while the Event scheduler is running. Once all events have finished, the BBS looks for the START key. If START is not pressed, the BBS will run NETCALL, otherwise the system will reset back to waiting for a call, ready to accept another call.

NETUPDT.CMD - this module is the opposite of **NETPREP**; it applies new packets that you have received to your message bases. **NETUPDT** is invoked automatically by the **NETCALL.CMD** after each message packet is downloaded from each board that you are calling. Once the packet has been processed, **INCOMING.DAT** is deleted and control is returned to **NETCALL** to continue the calling out process if additional boards still need to be called. This module may be run manually from the Dos Shell.

Setting up to PLACE ExpressNET! Calls

The real heart of the ***ExpressNET!*** system is the **NETCALL.CMD** module. It is this module that makes all of the outgoing calls to connect to other nodes in your ***ExpressNET!*** network.

The **NETCALL** command is automatically invoked after each run of the event scheduler; **DO NOT** place NETCALL in your Event Scheduler table. The BBS shell will automatically run NETCALL (a) after each caller hangs up, or (b) when the hour changes.

You control **NETCALL** through the **CALLOUT.DAT** file (edited with the command NETEDIT). You specify to NETCALL the following things:

o The phone number of each BBS to call.

o The baud rate of that BBS.

o How many seconds to wait for a carrier after dialing the number (just like you do in the Express! terminal program).

o Your network User Number and Password that was set up by the SysOp OF THE BOARD YOU ARE CALLING.

o What days to call this BBS (ie: Monday, Wednesday, and Friday).

o The maximum number of times to try to connect to this BBS.

o The "*call window*"; the hours that you want this board to be called. In other words, you can tell NETCALL to "call this BBS up to 255 times (max_retries), but only call between the hours of 1:00 am and 2:00 am".

o Where you would like the incoming messages from this BBS placed in your message bases. In other words, you can say "the messages that you get from message base 10 on this board, put into my message base 2".

Setting up to RECEIVE ExpressNET! Calls

You must inform BBS Express! Pro of each node number that you want to allow to receive messages from your BBS. You do this through (a) the **ECHONET.DAT** file and (b) your UserLog file.

The **ECHONET.DAT** file is used by NETPREP when it builds message packets (files) for each node. The **ECHONET.DAT** file informs NETPREP of the following things:

o Each node number to build a message packet for.

o The user number of the record for this node. NETPREP reads this record from your USERLOG file, uses the "Last read message" field so it knows which messages are new for this

node, and then rewrites the userlog record to reflect the change in the "last read message" field.

You must then add a new user through UEDITOR for each node that you want to receive messages from your BBS. There are 4 pieces of pertinent data that you must set in this new user record (all other fields are ignored).

These fields are:

o **User Handle:** place in here a string for your documentation. It's recommended that you put in something like "ExpressNET! node #45"

o **Real Name:** place the NODE NUMBER that this record is for in this field.

o **Password:** Set up a password for this node record. The caller will have to specify this password in his or her CALLOUT.DAT file, or the BBS will never let them on.

o **Message Security:** You must edit the **"Read security"** for this node record. For each message base that you want them to be able to receive, place a **'Y'** in the Read flag.

An Example ExpressNET! Setup

John and Bill would like to send and receive messages from each other's BBS's. John is node number **100**, Bill is node number **250** (your node number is your SERIAL NUMBER). Bill wants to receive messages from bases 1, 3, and 10 of John's BBS. But, he wants to put these messages into message bases 1, 2, and 3 on his BBS.

In this example, we are only going to go through the steps "one way"; that is, what would need to be done for Bill to receive messages from John. In reality, you would repeat these steps in reverse so that John could receive messages from Bill also (confused yet?).

Here are the required steps:

1. John goes into **UEDITOR** and **"A"**dds a new record. This new record is number 500. John does the following to this user record:

In "User handle:" places "ExpressNET! node #250"
"Real Name:" places "250"
"Password" places a password, let's say, "PRO1SYS"

Goes to **"Message Security"** and sets the following in the message base "read" flags:

```
----+----1----+----2----+----3--
y.y------y.....................
```

2. John uses **EDITFILE** to edit his **ECHONET.DAT** file. He places the following line in the **ECHONET.DAT** file:

```
250 500 Bill's node info
|   |   |
|   |   |--> solely for your documentation purposes
|   |------> the user record number for this node
|----------> the calling node number
```

3. John places **NETPREP** in his Event Scheduler as a timed event to be ran every day (if it's not already in there, that is). An ideal time to run this would be around midnight each day.

4. John runs the **SYSEDIT** program and makes sure that he has his Node information entered correctly.

5. John calls Bill and says, "*OK, Bill, your network user number on my BBS is 500, the password is PRO1SYS, and you are now set up to receive messages from my message bases 1, 3, and 10.*"

6. Bill runs the **NETEDIT** program and adds a new entry into his **CALLOUT** file. He keys in the following information:

```
  [A]  BBS Name : John's Node
  [B]  Number   : 804-744-8897
  [C]  Baud: 2400          [D] Seconds to wait: 45
  [E]  Usernum/PW:  500/PRO1SYS
  [F]  Days to call       S M T W T F S (call every day)
                          Y Y Y Y Y Y Y
  [G]  Call between hours 2 - 4 (call from 2:00 - 3:59 am)
  [H]  Maximum retries    100   (can try up to 100 times)
  [I]     Last tried                      :  0/00/00
          Last Connected                  :  0/00/00
          Last packet recv'd              :  0/00/00
  [L]  Incoming Message Redirection
----+----1----+----2----+----3-  --> (incoming bases)
         11111111122222222233
12245678931234567890123456789O1  --> (your bases)
| |      |
| |      |--> incoming base 10 to base 3
| |---------> incoming base  3 to base 2
|-----------> incoming base  1 to base 1
```

7. Bill runs the **SYSEDIT** program and makes sure that he has his Node information entered correctly.